Relatório T3 - Prolog com clpfd
Disciplina: Paradigmas de Programação
Alunos: Cainã Correa Caldas 18200415 e Vinicius Guedes Santos 17204690
Data: 5/12/2019
Universidade Federal de Santa Catarina

O programa tem múltiplos arquivos, mas todos são análogos, então apenas o wolkenkratzer.pl está comentado.

1: Explique o que você entendeu sobre programação de restrições, ilustrando como a mesma foi utilizada na solução do trabalho (exemplos de trechos e comentários a respeito do código fonte desenvolvido pelo grupo).

Programação orientada a restrições é bem parecido com o paradigma lógico. A impressão é que é um paradigma lógico mais inteligente. A biblioteca clpfd proporcionou a função maplist, e o operador 'ins'. A função maplist exigia que para toda a lista, a restrição passada como parâmetro fosse satisfeita. Enquanto isso, o operador 'ins' exigia que todos os elementos de uma lista estivessem dentro certa faixa de valores. Confira o trecho:

```
[...]
append(Rows, Vs), Vs ins 1..4,
maplist(all_distinct, Rows),
[...]
```

Além disso, um operador da biblioteca foi utilizado em uma regra recursiva. Foi o operador "#>", que exigia que o que estivesse a esquerda fosse maior do que o que estivesse à sua direita. O operador '#<' funciona de maneira análoga. Confira o trecho:

```
[...]
quantosVejoRecursivo([], _, 0).
quantosVejoRecursivo([H|T], Max, V) :- H #> Max, quantosVejoRecursivo(T, H, V1), V is V1 + 1.
quantosVejoRecursivo([H|T], Max, V1) :- H #< Max, quantosVejoRecursivo(T, Max, V1).
[...]
```

2: Indique possíveis vantagens e desvantagens que o grupo identificou no uso de programação de restrições para resolver o problema.

Percebemos que o código executou espantosamente rápido em relação às implementações do mesmo problema em Haskell e Lisp. Além disso, ele ficou muito mais enxuto. Isso ajuda em manutenibilidade e torna o programa mais fácil de ser compreendido. Por outro lado, a maioria dos programas desenvolvidos hoje em dia utilizam a POO, então teríamos que aprender ainda como utilizar a programação orientada a restrições dentro de um programa em POO, o que fiquei feliz em ler rapidamente na internet que é possível.

3: Destaque como o usuário poderÁ informar a entrada e de que forma o resultado é apresentado para o usuário.

O usuário deve fazer a seguinte consulta:
problem(1, Rows), wolkenkratzer(Rows), maplist(portray_clause, Rows).

Porém, considera-se que ele já carregou o arquivo correto no swipl. Pois existe um arquivo para matrizes 4x4, um para 5x5 e um para 6x6.
Além disso, o usuário deve entrar no e fazer alguns ajustes:

append(Rows, Vs), Vs ins 1..6,
Na linha acima, manter 1..X quando os números permitidos vao de 1 a X, e trocar para 0..(X-1) caso a matriz aceite espaços.

% Linhas da esquerda para a direita
vejoCerto([A11, A12, A13, A14, A15, A16], 2),

O programa contém cinco trechos com os comentários: [% Linhas da esquerda para a direita, % Colunas de cima para baixo, % Linhas da direita para a esquerda, % Colunas de baixo para cima, % Diagonais]. Tenha em mente que a regra 'vejoCerto' estará colocando a restrição de que a lista deve ver, da esquerda para a direita, exatamente a quantidade de prédios passada no segundo parâmetro. Com isso, adaptamos para as outras direções através de colocar os índices na ordem correspondente. Então, no exemplo acima, observa-se que os índices corresponde a posições que estão indo da esquerda para a direita mesmo, e o que se espera é que seja possível ver 2 prédios. Vejamos outro exemplo agora:

% Colunas de baixo para cima
vejoCerto([A41, A31, A21, A11], 0),

Nesse caso, os índices estão sugerindo que a lista seja lida de baixo para cima!

O usuário deve, portanto, comentar os trechos desnecessários para aquele tabuleiro, e fornecer o segundo parâmetro para cada uma dessas funções, restringindo a quantidade de prédios para ser vista daquele ponto de vista.

O resultado é apresentado com uma simples impressão do tabuleiro resolvido na tela.

4: Comente as dificuldades encontradas e as soluções adotadas pelo grupo.
Tivemos dificuldade em entender como a clpfd funciona por baixo dos panos, mas usá-la foi relativamente fácil. Tivemos um problema em que colocamos uma lógica, porém a consulta retornava 'false'. Resolvemos isso através de colocar outra lógica. Nosso aprendizado da biblioteca foi baseado em exemplos, principalmente, mas também video aulas.

5: Faça uma breve comparação entre o paradigma lógico e o paradigma funcional para resolver o problema proposto.

O paradigma lógico parece ser bem mais adequado ao problema porque ao invés de iterar sobre várias listas, criar várias funções, pudemos criar poucas regras, bem mais enxutas, e no final o programa ficou absolutamente mais eficiente. Parece que ao utilizar o funcional, a probabilidade de produzir um código menos eficiente é altíssima, até porque é o programador que implementa o backtracking, enquanto no prolog, isso se resolve pelo compilador e em tempo de execução.